Computational Models

For human behavior representation Interchange Standard

Avelino J. Gonzalez
Jose Castro
Robert Franceschini
University of Central Florida

Advantages of Common Interchange System

- Knowledge can be captured and represented only once.
- Knowledge core is common - only the representation varies.
- Cumulative base of knowledge can be built over time.

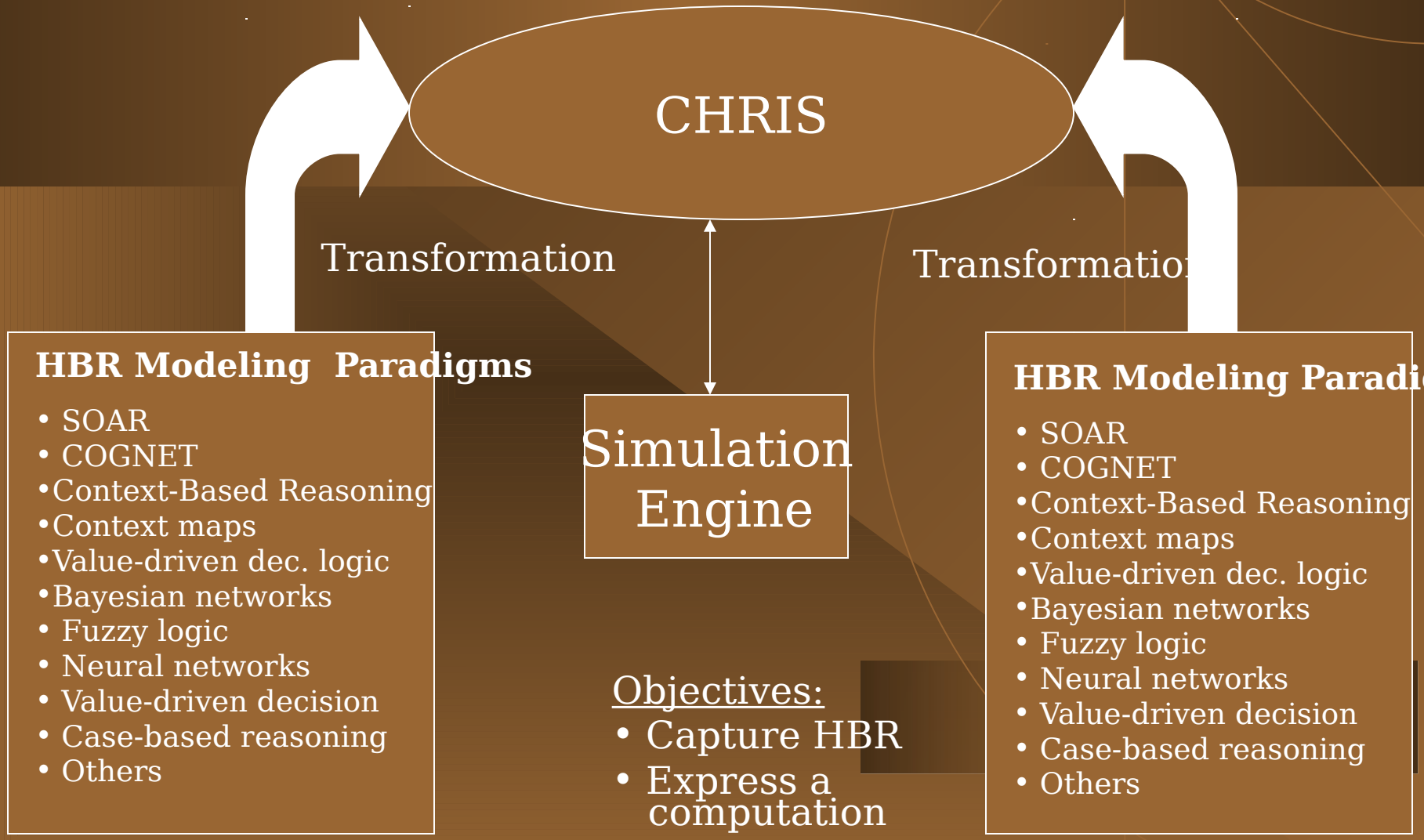
Human Behavior Knowledge

- Temporal considerations of great importance - short time scale
- Highly individualized
- Highly dependent on context
- Grows with experience
- Not completely understood

Terrain Data - SEDRIS

- Temporal considerations of less importance
- Knowledge not involved
- Commonly perceived
- Does not grow with experience
- Not context dependent
- Largely understood for today's application

Notional CHRIS



Common Human Behavior Interchange System Requirements

- Must select simple yet powerful paradigm as basis.
- Paradigm can grow without invalidating prior knowledge representations.
- Existing representations must be transformable to paradigms.
- File format must be developed.

Turing Machine

- Most powerful computational paradigm
- Reads and writes on an infinite tape memory with symbols
- It has two functions, a state transition function and a tape action function
  - tape action can be:
    - move left
    - move right
    - write symbol

Turing Machines

- Can recognize anything computable
  - The most general paradigm
  - Generality hinders applicability
  - Of mathematical/theoretical interest only
  - Have never been implemented for actual computer use

Proposed Paradigm

- Finite State Machine

Finite State Machine Characteristics

- Mathematical Abstraction of the form
- FSM $\equiv (\Sigma, Q, R, F, \delta)$
  - Σ possible events
  - Q a set of states
  - R a start state
  - $F \subseteq Q$, set of accepting states
  - $\delta : Q \times \Sigma \rightarrow Q$, a state transition function

Properties of FSM

- FSM recognize regular languages (languages denoted by regular expressions):
  - $\{a|b\}^{*}c^{+}$
  - Can be used, and are frequently used for computation in:
    - Compilers
    - Simulation systems
    - Text search systems

FSM Characteristics - Advantages

- Mathematically sound and well-studied abstraction
- Can be represented as Graphs or Tables
- Easy to implement
- Make for efficient computer systems
- Represent behavior (action-reaction) naturally

FSM Characteristics - Advantages (Cont.)

- Already used in several CGF applications (e.g., ModSAF CCTT and others)
- Several extensions exist that make them powerful representational paradigms

FSM Characteristics - Disadvantages

- Do not have unlimited memory, therefore, cannot solve certain kinds of problems
- Encourage a “behaviorist” model and not a “cognitive” model, (emphasize action-reaction and not internal representation)
- Limited complexity of problems able to solve.

FSM Characteristics - Disadvantages (Cont.)

- Are not by themselves a Interchange format.

Variations of Finite State Machines

- There are variations of finite state machines that address these disadvantages
  - Non-deterministic FSM
  - Push-down Automata
  - Hybrid Automata
  - Cellular Automata
  - Fuzzy Automata
  - Timed Automata
  - Tree Automata

Non-deterministic Finite State Machines (NFSM)

- Automata whose state transition function δ evaluates to set of possible states.
- Has same computing power as FSM ==> no computational advantage
- Can be a more “natural” representation

Push Down Automata

- An FSM with a stack - permits push and pop from memory.
  - The state transition function δ depends on the value on top of the stack and on the state
  - Function added to define the push and pop operations
    - (state,TOS,input) =>action(TOS)

Pushdown Automata

- Recognize nested contexts
- Are more powerful than FSM (If a FSM exists to recognize situation L, then a Pushdown automata also exists)
- Used mostly in computer language compilers

Hybrid Automata

- Combines time-driven and event driven dynamics.
- A generalization of a finite-state automaton, equipped with a set of variables
- Is able to model discrete events and continuous activities, governed by a set of differential equations.
- Provides a modeling paradigm for hybrid systems
- Described by a finite set of real-valued variables and a labeled multi-graph (V,E)

Hybrid Automata

Formally: $H = (X, V, E, syn, act, inv)$

- Continuous variable set $X = (x_1, x_2, \ldots, x_n)$

  Valuation of H $s = (x_1 = a_1, x_2 = a_2, \ldots, x_n = a_n)$
- Control locations ***V***

  A state of the automaton is a pair **(v,s).** Where $v \in V$ is the location and $s \in R^n$ is a valuation

Hybrid Automata

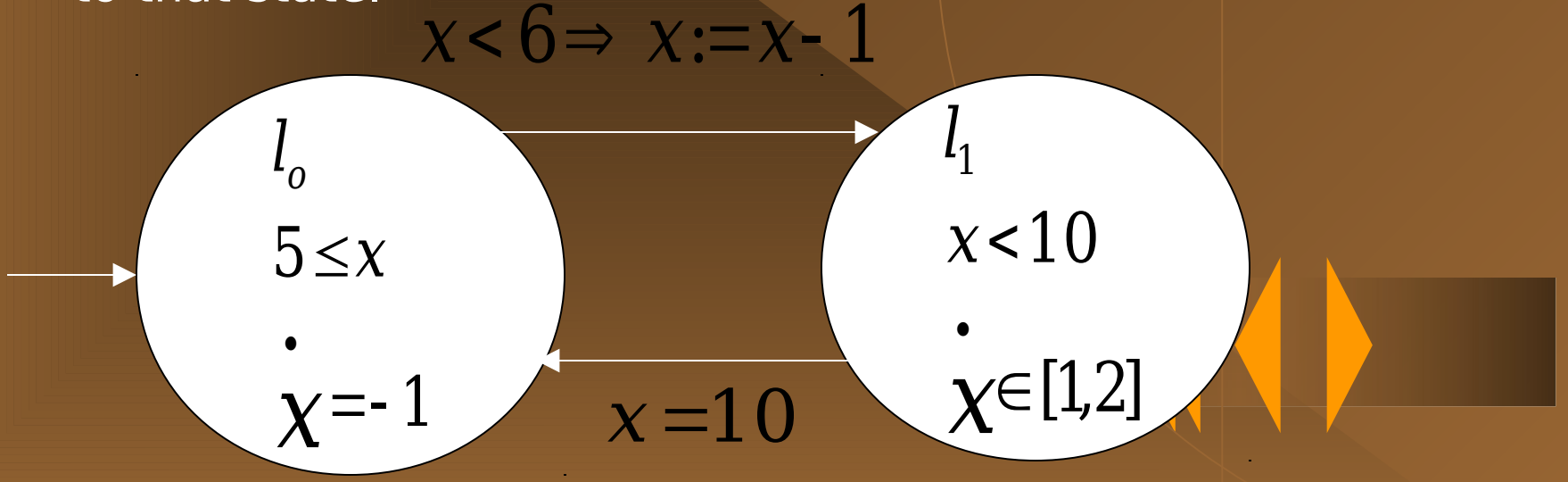
- Finite set of transitions **E**
- A transition $e = (v, a, u, v')$
- Source location $v \in V$
- Target location $v' \in V$
- Synchronization label $a \in syn$
- Transition relation $u \subseteq S^2$

Hybrid Automata

- Synchronization labels **syn**, parallel composition of two automata.
- Activities **act**, assigns to each location $v \in V$ a set of activities
- A function invariant **inv,** assigns to each location $v \in V$ an invariant $inv(v) \subseteq s$

Hybrid System

- A node in the automaton is a discrete state combined with the continuous dynamics connected to that state.



SHIFT

- The name SHIFT is a permutation of HSTIF (Hybrid System Tool Interchange Format)
- Description language for dynamic networks of hybrid system.
- System consists of components, which can be created, interconnected and destroyed as the system evolves in time.
- Components exhibit hybrid behavior, consisting of continuous-time phases separated by discrete event transitions.

Dymola

- **Dynamic Modeling Language**
- Object-oriented modeling
- Reuse of library models
- Graphical model composition
- Symbolic equation processing
- Efficient hybrid simulation including 3-D animation
- Manipulate Hybrid Models

Cellular Automata

- Unidimensional or bidimensional grid of cells
- Each one of the cells is a simple automaton
- The automata’s input is the state of the neighboring cells
- Used in biological and population growth mathematical studies
- Turing equivalent

Comparison

- FSM ≡ NFSM < Pushdown Automata < Turing Machines
- FSM used most of all
- NFSM used for notation
- Pushdown automata used in compilers only
- Turing machines never used

FSM for Human Behavior

- Mathematically sound and proven
- Already used in many systems
- Simple and efficient
- Even though not a file format in itself, can be easily stored and transmitted
- A working core of FSM representation of HB guaranteed (already exists).

FSM for Human Behavior

- Information in other knowledge forms can be transformed to FSM
  - Transformation from case-based and context-based systems is straight forward for extended automata
  - Transformation from rule-based paradigms is also direct
  - Connectionist Models?

FSM and Neural Networks

- ANN’s used for automatic knowledge extraction
- Can learn from examples
- Recurrent neural networks (RNN) easily mapped to FSM and vice-versa
  - RNN have naturally unbounded input lengths.

Suggested Approach

- Feasibility Study - using Finite State Automata extensions as the interchange standard.
- Study feasibility of Transforming between the several paradigms into the FSM standard.
- Develop prototype system to evaluate effectiveness.